Processing Graph Method Tool (PGMT) Installation Manual

by
Wendell L. Anderson
October 31, 2002

Introduction:

This report describes the steps necessary to install the Processing Graph Method Tool (PGMT) onto a system of computers operating in a UNIX and/or LINUX environment. PGMT is an implementation of the PGM specification that includes the software necessary to run a data flow graph on a collection of processors. The tool contains a Graphical User Interface for creating, saving, and translating graphs into C++ code; libraries to support the linking and run-time execution of a graph; debugging aids; and a tool for generating the code necessary to transfer user-defined data types between computers using the Message Passing Interface (MPI) standard.

This document has been written under the assumption that the user is familiar with UNIX commands, directory structures, and XWindows, but makes no assumptions about the user's knowledge of the Processing Graph Method (PGM). Instructions for using the tool to develop programs can be found in the PGMT User's manual. PGMT has been successfully installed on Suns, SGI's, and Linux-based PC's as well as members of the SGI Origin supercomputers.

The software has been developed using g++ (GNU C++) and Java 2. PGMT uses the Message Passing Interface (MPI) for inter-processor communication. PGMT is released under the GNU public license and is available on CD-ROM by sending e-mail request to pgmt-support@ait.nrl.navy.mil or a request in writing to

Commanding Officer
Attn: Code 5580
Naval Research Laboratory
4555 Overlook Avenue
Washington, D. C. 20375

PGMT may also be downloaded over the internet from the site pgmt.ait.nrl.mil. The distribution consists of the seven files listed in Table 1.

COPYING	GNU General Public License
DOC_LIC	GNU Free Documentation License
SUPPORT.pdf	PGMT documentation in PDF format

CONTRIB.pdf	list of contributors in PSF format
PGM2.tgz	gzip tar ball of the documentation
PGM2_DOC.tgz	gzip tar ball of PGM2
PGM2_SUP.tgz	gzip tar ball of ancillary software

Table 1: Distribution Files

The DOC_LIC and COPYING files outline the conditions of the GNU Free Software Foundation on the use and distribution of PMGT and should be read before installing the software. The two pdf files can be read and printed using the acrobat reader available from http://www.adobe.com. These files provide the names and contact information of the members of the team who developed PGMT as well as copies of their papers and manuals.

The three gzip tar balls contain the PGM2 software, free third party software used by the tool, and a second copy of the documentation. The support software package is only needed if some of the software mentioned under systems requirements is not already resident on the system or javaHelp is not part of the java installed java. The support software included in the distribution is listed in Table 2

Software	File	Size (MB)
g++ 2.95.2	gcc-2.95.2.tar.gz	12
g++ library patch	glibc-2.2.patch	<1
Java 1.3	j2sdk-1_3_1_03-linux-i386-rpm.bin	25
JavaHelp	javahelp1_1-unix-no_rt.bin	8
MPICH C++ bindings	mpi2c++.tar.gz	<1
MPICH 1.2.1	mpich-1.2.1.tar.gz	9

Table 2: Support Software

Throughout these instructions, whenever the user is asked to enter data from a keyboard, items in **bold** should be typed exactly as they appear while items in *italics* are user dependent. Whenever the instructions indicate a line is to be entered a carriage return is entered after the typing of the line. For example, if the user is asked to enter the line

cd *full_directory_name*

the user should type cd, the actual directory name, and the enter key.

System Requirements

PGMT has been developed using the GNU g++ 2.95.2 compiler and java 2. For passing data between processors, the tool uses the Advanced Programming Interface (API) of the Message Passing Interface (MPI) and has been developed using the MPICH implementation of MPI. The installation procedure uses GNU's gmake utility to build the libraries and executables of PGMT. The PGMT Graphical User Interface (GUI) uses the javaHelp libraries to supply on line help for the GUI.

Before installing PGMT on a network of processors, the user should verify that the operating system directories for the processors have g++ 2.95.2, gmake, java 2, and an implementation of MPI installed. If all of this software is not available then the user must have the system administrator install the missing software on the operating systems. In the case of heterogeneous systems, or systems with processors running different versions of the operating system, a copy of the software must be available for each system. Software packages for installing the previously mentioned software are available as part of the distribution.

Since in the case of installation on a PC LINUX system, the user is often also the system administrator, instructions are contained in Appendix A for the installation of the support software on a RedHat LINUX system.

When installing PGMT on a High Performance Computer (such as an SGI Origin 3000 series), the user should use the native MPI implementation. Since this implementation has been tuned to the specific platform, using the native MPI results in a more efficient operation of the PGMT code.

PGMT Installation

After the user has copied the PGMT distribution tar balls to a disk directory, the required tar balls should be unzipped and unbarred using the commands

```
gunzip PGM2.tgz
tar -xvf PGM2.tar
```

These commands will put installation software into the directory PGM2 residing in the current directory.

After the files have been extracted, the user should change his directory to the directory **PGM2/config** and create a new configuration file **pgm2**_*system*.**csh.** A template of the configuration file is contained in **pgm2_template.csh**. The configuration file consists of four parts: definitions needed to build and run the basic PGMT application, those needed for using the debugging portion of PGMT, those needed for building the tool for using user-defined data types, and additions to the path and aliases.

The first part of the configuration file defines the environmental variables used by PGMT (Table 3).

PGM2_HOME	PGMT root directory
PGM2_MPI	MPI root directory
PGM2_TYPE	Interconnection type
PGM2_HOST	Name of Host
PGM2_CC	C compiler to use
PGM2_CCC	C++ compiler to use
PGM2_LDLIBS	Libraries for linking
PGM2_LDFLGS	Linker flag(s)
PGM2_GETTIME	Flag for presence of gettime

Table 3: PGMT environmental variables

PGM2_HOME is set to the full path name for the directory containing the PGM2 directory. **PGM2_MPI** is the directory that contains the implementation of MPI used by PGMT. **PGM2_HOST** is the name of the current host computer. **PGM2_CC** and **PGM2_CCC** are the names of the C and C++ compilers and may also include default switch specifications. For implementations using MPICH these variables will begin with **mpicc** and **mpiCC** respectively. **PGM2_LDFLAGS** and **PGM2_LDLIBS** specify default flags and libraries respectively for the C++ linkers. Finally, **PGM2_GETTIME** is a flag indicating whether or not the function gettime is part of the system software. While gettime is usually found in UNIX systems, Linux systems usually do not have it. When it is not present, PGMT provides a version.

The second part of the .csh file sets up the environmental variables for running PGMT in a debug mode. The environmental variables **PGM2_CC** and **PGM2_CCC** are redefined so that the -DDEBUG switch is used when the PGMT code is compiled ensuring that the debug code is included in the executable for

the program. The other environmental variables are set to determine at run-time the set of debug statements to be printed out. These lines are commented out in the template version of the .csh file and should be commented out in the .csh file created by the user.

The third part of the .csh files contains the definitions of the variables required to build and test C++2MPI (mtool) the software that allows the user to create his own data types without worrying about the details of MPI data types. Further explanations of these environmental variables can be found in the C++2MPI integration manual.

Finally in the last part of the configuration file, the path is updated so that it contains the directories of the PGMT installation executables and aliases are established for the GUI and gmake.

Examples of pgm2_*system*.csh files for a SUN workstation, an SGI workstation, a Red Hat Linux PC, and an HPC system are can be found in the directory **PGM2/config.**

Once the .csh has been created, it should be invoked by the command

source *${PGM2_HOME}/***config/pgm2**_*system*.**csh**

In order to avoid typing this command every time the user logs in, this command line can be included in the user's .cshrc file.

Since the GUI uses javaHelp, this software must be installed if it is not part of the Java resident on the system. To install javaHelp, unzip and untar the support software by entering the commands

gunzip PGM2_SUP.tgz
tar -xvf PGM2_SUP.tar

and then change the working direct to the support sub-directory by issuing the command

cd ${PGM2_HOME}/support

Next install javaHelp in the PGM2 directory by entering the command

javahelp1_1-unix-no_rt.bin

The program asks the user to select a java virtual machine by typing a number. Type **1** and click on the displayed window. Click the **Next** button until a window asking for the installation directory appears. Insert **PGM2/** before **jh1.1** in the path name displayed and click ok. javaHelp will now be installed.

After javaHelp has been installed, the user should install the PGMT GUI by entering the commands

```
cd ${PGM2_HOME}/src/GUI
make
make install
make cleanup
```

The next step is to install C++2MPI, the software for generating MPI code for user defined data types. This is accomplished by executing the following commands

```
cd ${PGM2_HOME}/src/mtool
make
make install
make cleanup
```

In order to verify that mtool (C++2MPI) has been properly built and installed, the tests provided in the Makefile in **${PGM2_HOME}/src/mtool** should be executed**.** The tests are run with the command

```
make test
```

The expected results from these tests are given in Appendix B of this manual. Significant differences should be reported to pgmt-support@ait.nrl.navy.mil.

Testing the PGMT Installation

To verify that the PGMT software has been properly installed and configured, the user should build and run the Sort and TestDataType programs. The program Sort sorts a list of 32 floating point numbers and prints them out in ascending order. The TestDataType program reads and writes a variable of type phone (a user defined data type) to a queue in a graph.

To run either of the tests, the user must first log into the system of interest and execute the configuration file by using the commands

source ${PGM2_HOME}/config/pgm2_*system*.csh

To build and run the Sort program the user enters the commands

```
cd ${PGM2_HOME}/apps/Sort
make
run
```

If all the software has been installed properly, 32 lines of the form

Out nn **is** fffff

(where nn ranges from 0 to 31 and fffff are floating point numbers in increasing numerical order) will be printed to the screen.

To build and run the TestDataType program the user enters the commands

```
cd ${PGM2_HOME}/apps/TestDataType
make
run
```

If the software has been properly installed then the two lines

```
Onoff = 1.0 Phase = 0.5
Onoff = 1.0 Phase = 0.5
```

will be printed out

References

1. Anderson, W., " Processing Graph Method Tool (PGMT) Installation Manual" October, 2002
2. Anderson, W., " Processing Graph Method Tool (PGMT) User's Manual", October, 2002
3. Bharadwaj, K. and Stevens, R., "The Graph State File Specification", May, 2001
4. Boroujerdi, Ali, "Scheduler Documentation" ,June, 2001
5. Collins, J. "An Approach to Scheduling Task Graphs with Contention in Communications", June, 2001
6. Collins, J. "Scheduling in PGMT", April, 2002
7. Hillson, R., "C++2MPI/PGMT Integration". July, 2002

8. Hillson R. and Iglewski, M. C++2MPI: A Software Tool for Automatically Generating Datatypes from C++ Classes", IEEE Proceedings of the International Conference on Parallel Computing in Electrical Engineering (PARALEC 2000), 27-30 August 2000, Trois-Rivieres Quebec CA, pp. 13-17
9. Iglewski, M. , "Processing Graph Method - GUI Handbook", July, 2002
10. Kaplan, D., "An Introduction to the Processing Graph Method", March, 1997
11. Kaplan, D., “PGM Documents Synopses”, October, 2002
12. Kaplan, D. and Stevens R. , " "Processing Graph Method 2.1 Semantics", July, 2002
13. Scannell, C. "GramGraph Graphical User Interface", March, 2002
14. Scannell, C. "Interfacing the Command Program to a PGM Graph", June, 2002
15. Stevens. R., "Documentation of the Graph Class Library (GCL)", November, 2001
16. Stevens, R. , "Draft Notes for GUI Forms", July, 2002
17. Stevens, R. , "Handbook for Authors of Transition Statements", July, 2002
18. Stevens, R. , "Distributed Processing in PGMT", August, 2002
19. Stevens, R. , "Specification for the Translator Output", January, 2002

APPENDIX A
PGMT Support Installation Under Red Hat Linux

This appendix contains a description of the installation of the support software required by PGMT on Red Hat 8 LINUX system. After completion of this section the user should be able to install and successfully test the PGMT software. If problems arise during the installation or testing that the installer cannot resolve, send e-mail to pgmt-support@ait.nrl.navy.mil.

The RedHat LINUX software is available on CR-ROM or can be downloaded from http://www.redhat.com/download/mirror.html.

The user should log into the root account and copy PGM2_SUP.tgz into the /source directory. To install g++ and mpich the user should perform the following steps.

1.) Untar the gcc

```
/bin/mkdir --verbose -p /source/gcc-2.95.2_linux
cd /source
tar xvf gcc-2.95.2.tar.gz
```

2) Patch the gcc source

```
cd /source
patch -p0 < glibc-2.2.patch
```

3) Build gcc binaries

```
cd /source/gcc-2.95.2_linux
../gcc-2.95.2/configure --prefix=/usr/local/gcc-2.95.2
make
make install
```

4) Compile a gcc-2.95.2 version of mpich-1.2.1

```
cd /source
tar xzf mpich-1.2.1.tar.gz
cd /source/mpich-1.2.1/
./configure -prefix=/usr/local/gcc-2.95.2 \
      -cc=/usr/local/gcc-2.95.2/bin/gcc \
      -c++=/usr/local/gcc-2.95.2/bin/g++
make
make install
make clean
```

APPENDIX B

This appendix contains the output that should be obtained from the mtool tests for the mtool utility. If there are significant differences between the output obtained by running the mtool test and this output, send e-mail with the output to pgmt-support@ait.nrl.navy.mil.

```
test1: sending a variable of a simple class
ex1 successfully completed

test2: complete example prepared by Roger Hillson
StatusProbe.MPI_Source = 1
StatusProbe.MPI_Tag    = 99
StatusProbe.MPI_Error  = 0
Received:  x1:  1 x2:  4  z: 12.340000
StatusProbe.MPI_Source = 2
StatusProbe.MPI_Tag    = 99
StatusProbe.MPI_Error  = 0
Received:  x1:  2 x2:  4  z: 24.680000
StatusProbe.MPI_Source = 3
StatusProbe.MPI_Tag    = 99
StatusProbe.MPI_Error  = 0
Received:  x1:  3 x2:  4  z: 37.020000

Send class instance report_s: rep.x1 1 rep.x2 4 rep.z 12.340000

Send class instance report_s: rep.x1 3 rep.x2 4 rep.z 37.020000

Send class instance report_s: rep.x1 2 rep.x2 4 rep.z 24.680000

test3: sending a variable of a class containing a field of another class
ex3 successfully completed

test4: sending a variable of a derived class
ex4 successfully completed

test5:sending a variable of a derived class with virtual functions
ex5 successfully completed

test6: sending a variable of a templated class
ex6 successfully completed
```

test7: sending a structure with a private member
ex7 successfully completed

test8: sending a variable of a templated class with templated base
ex8 successfully completed

test9: sending a variable of a local class
ex9 successfully completed

test10: processing more than one file
ex10 successfully completed

test11: processing a file with a pointer typedef
gmake[2]: [all] Error 1 (ignored)
***** pointer component detected

test12: processing a file with a class containing a pointer var
gmake[2]: [all] Error 1 (ignored)
***** pointer component detected

test14: processing a file with a base class containing a pointer var
gmake[2]: [all] Error 1 (ignored)
***** pointer component detected

test15:sending a var of a templated class with an arg of a templated class
ex15 successfully completed
P4 procgroup file is ../machineFile.

test16: sending a var of a templated class with an arg being a value:
ex16 successfully completed

test17: checking multiple calls of AIT_build_T_MPI_datatype routine
ex17 completed successfully

test18: checking adding existing symbols to *.a
ex17 completed successfully
P4 procgroup file is ../machineFile.
ex17 completed successfully

test19:
ex19 successfully completed

test20: sending a var of a class with members of predefined types
ex20 successfully completed.

test21:
ex21 successfully completed.

test22:sending a variable of non-templated class with templated base
ex22 successfully completed.

test23:
ex23 successfully completed

test24: checking whether the same handles for duplicate typemaps
ex24 successfully completed
ex24 successfully completed
ex24 successfully completed
ex24 successfully completed

test25:sending a var of a templated class with an arg of a templated class
ex25 successfully completed

test26: checking non-default output file names
ex26 successfully completed

test27: source files in different directories
ex27 successfully completed

test28: sending a value of a typedef not being a class
ex28 successfully completed

MyList.h contains the following prototypes: complex.h fcomplex.h.
There are user-specified leaf types.
Build both the default and derived MPI datatypes.
Process 0 constructs complex variables.
Process 3 constructs complex variables.
Process 1 constructs complex variables.
Process 2 constructs complex variables.
3: Pass S/R test for float variable.
receive complex variable on 3
3: Pass S/R test for complex variable.
1: Pass S/R test for float variable.
receive complex variable on 1

1: Pass S/R test for complex variable.
2: Pass S/R test for float variable.
receive complex variable on 2
2: Pass S/R test for complex variable.